HOWARD R. HUGHES College of ENGINEERING

DASL-100.2 C++ Programming and Linux

Drones and Autonomous Systems Lab @ UNLV

Week 2-2

1. Array.
2. String.
3. Pointers.

Instructor: Truc Tran

02-05-2023

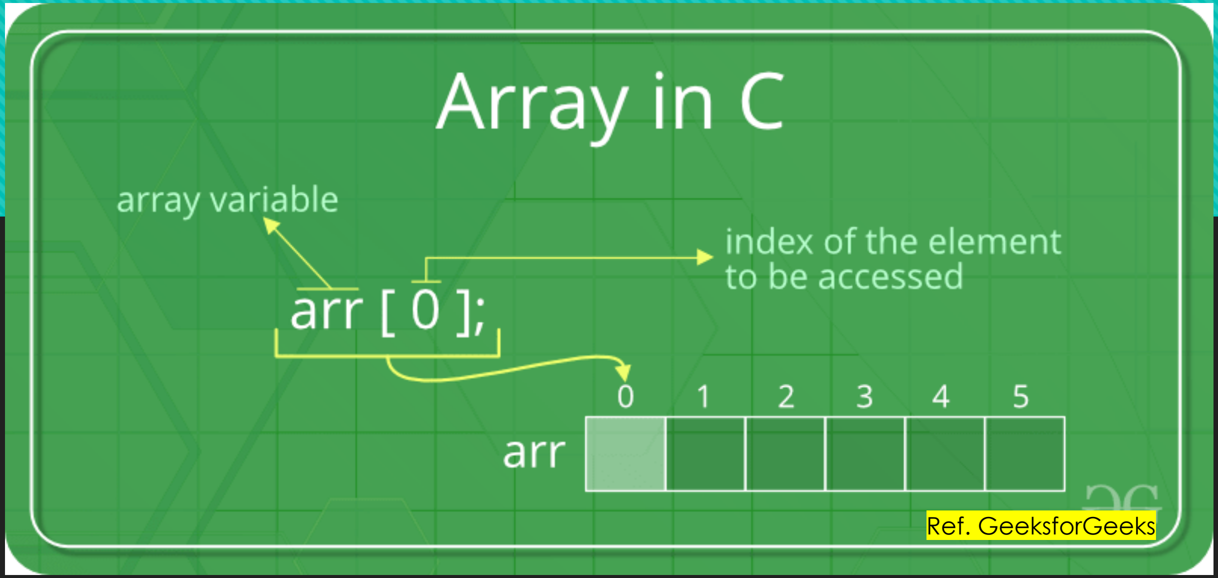
UNLV Howard R. Hughes College of Engineering

DASL-100.2 C++ Programming and Linux

DASL Drones and Autonomous Systems Lab @ UNLV

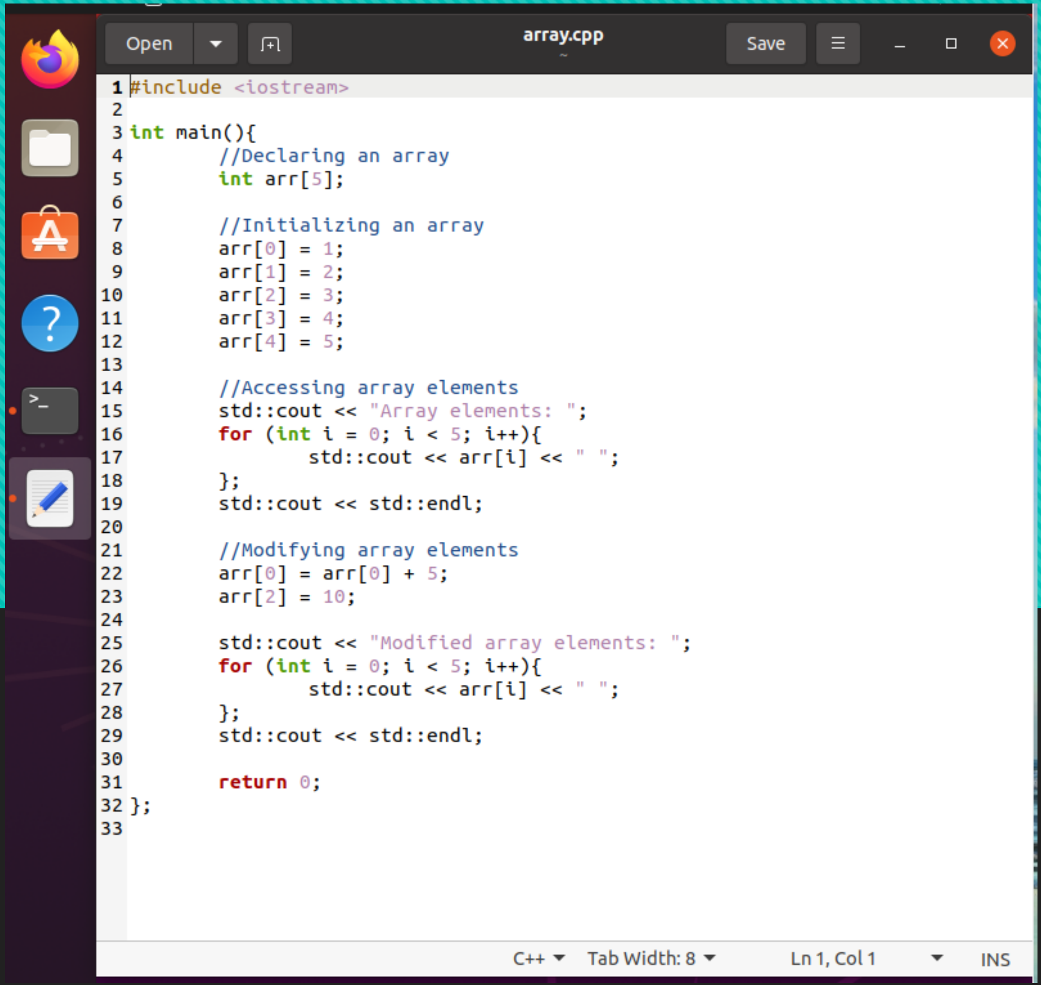
1. Array

- In C++, an array is a collection of elements of the same data type stored in contiguous memory locations. Arrays are declared with a specified size and data type, and can be indexed to access individual elements.
- C++ arrays can have any number of dimensions, but the most common ones are 1D, 2D, 3D, and sometimes 4D arrays. Example:
    - 1D: int arr[3];
    - 2D: int arr[3][3];
    - 3D: int arr[3][3][3];
    - 4D: int arr[3][3][3][3];



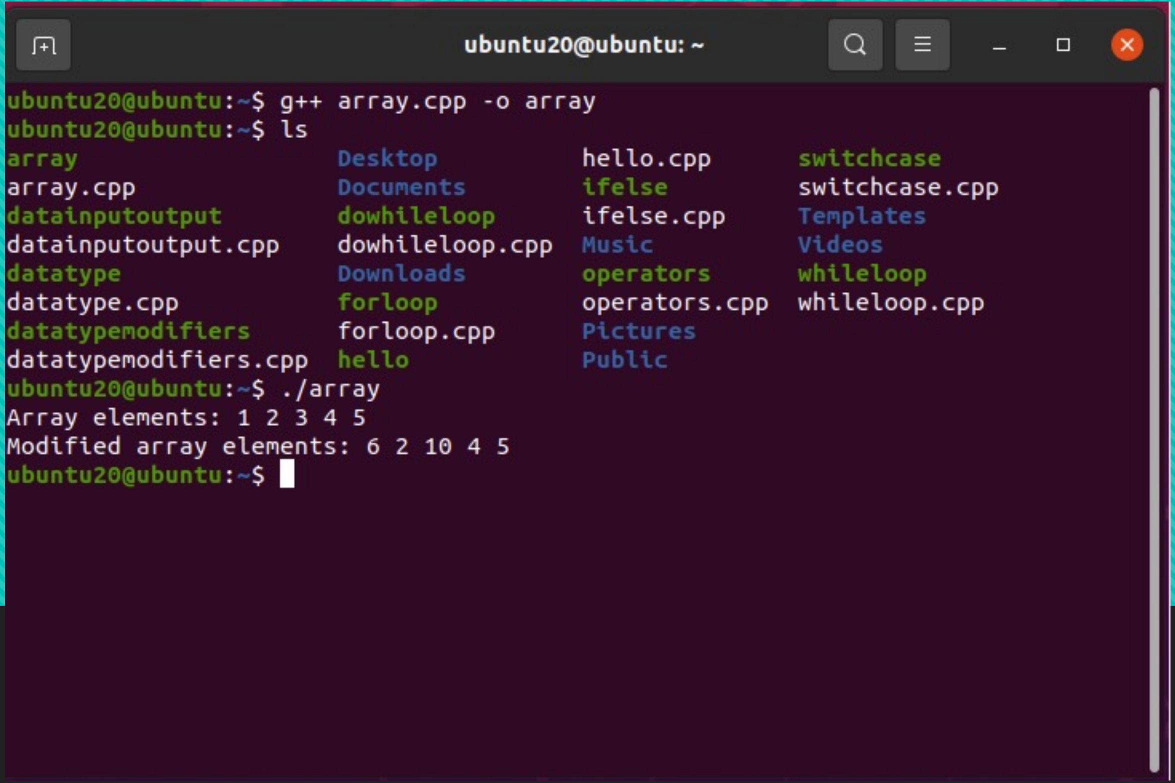
DASL-100.2 C++ Programming and Linux

1. Array



array.cpp

```cpp
#include <iostream>

int main(){
	//Declaring an array
	int arr[5];

	//Initializing an array
	arr[0] = 1;
	arr[1] = 2;
	arr[2] = 3;
	arr[3] = 4;
	arr[4] = 5;

	//Accessing array elements
	std::cout << "Array elements: ";
	for (int i = 0; i < 5; i++){
		std::cout << arr[i] << " ";
	};
	std::cout << std::endl;

	//Modifying array elements
	arr[0] = arr[0] + 5;
	arr[2] = 10;

	std::cout << "Modified array elements: ";
	for (int i = 0; i < 5; i++){
		std::cout << arr[i] << " ";
	};
	std::cout << std::endl;

	return 0;
};
```



ubuntu20@ubuntu: ~

```
ubuntu20@ubuntu:~$ g++ array.cpp -o array
ubuntu20@ubuntu:~$ ls
array                    Desktop          hello.cpp      switchcase
array.cpp                Documents        ifelse         switchcase.cpp
datainputoutput          dowhileloop      ifelse.cpp     Templates
datainputoutput.cpp      dowhileloop.cpp  Music          Videos
datatype                 Downloads        operators      whileloop
datatype.cpp             forloop          operators.cpp  whileloop.cpp
datatypemodifiers        forloop.cpp      Pictures
datatypemodifiers.cpp    hello            Public
ubuntu20@ubuntu:~$ ./array
Array elements: 1 2 3 4 5
Modified array elements: 6 2 10 4 5
ubuntu20@ubuntu:~$
```

2. String

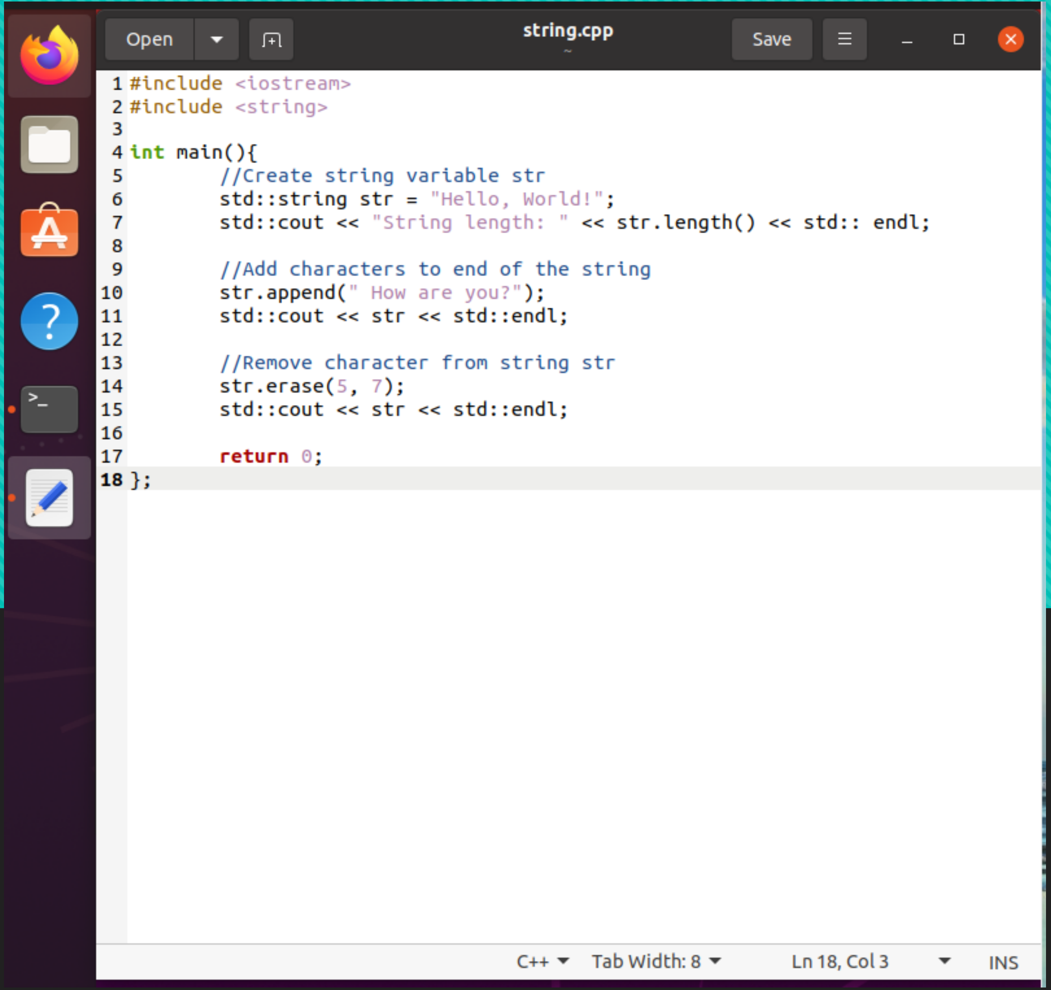
- In C++, string is a data type that represents a sequence of characters.
- The **std::string** class in C++ provides a convenient way of manipulating strings of characters. It is defined in the <string> header file, which is part of the Standard Template Library (STL).
- One of the main advantages of using std::string is that it automatically manages the memory required to store the string, freeing the programmer from having to manually allocate and deallocate memory for strings. In addition, **std::string** provides many useful methods for working with strings, such as:
  - length(): returns the number of characters in the string
  - empty(): returns a boolean indicating whether the string is empty or not
  - clear(): removes all characters from the string
  - append(): adds characters to the end of the string
  - insert(): adds characters at a specified position in the string
  - erase(): removes characters from the string
  - substr(): returns a substring of the original string

Ref. GeeksforGeeks

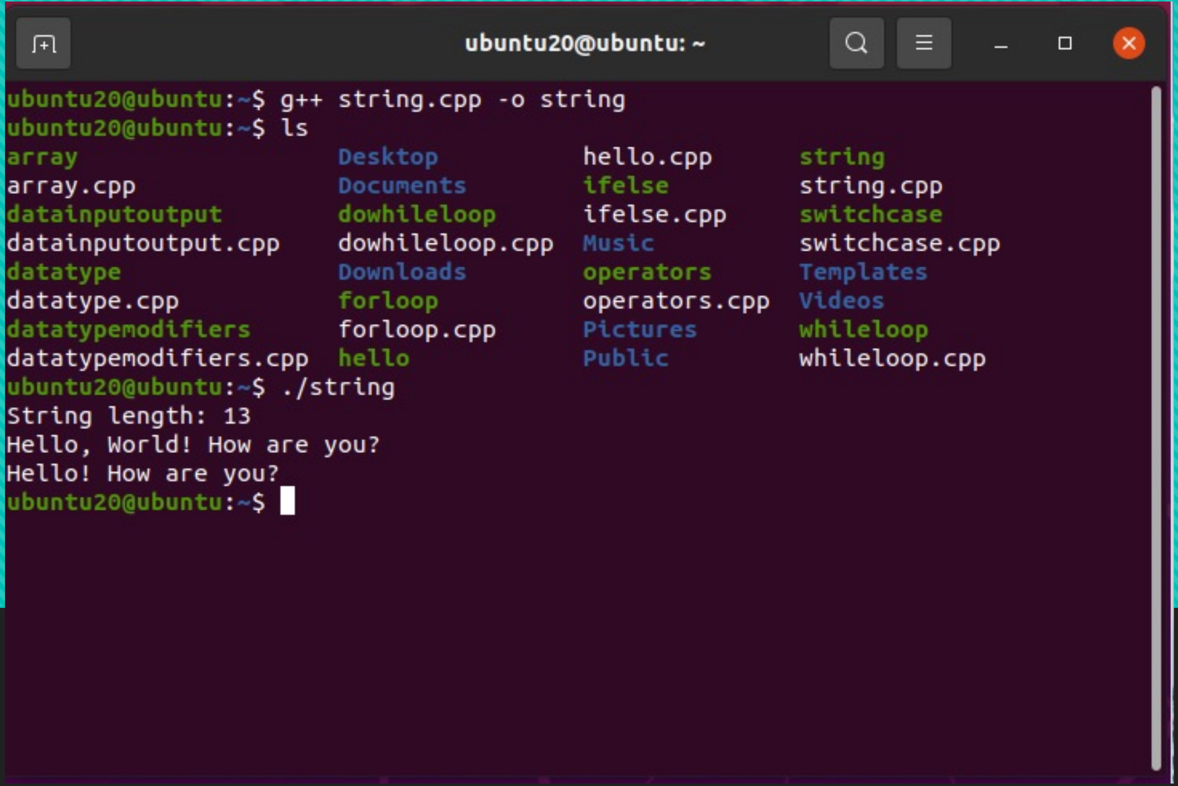
DASL-100.2 C++ Programming and Linux

Drones and Autonomous Systems Lab @ UNLV

2. String



```cpp
#include <iostream>
#include <string>

int main(){
        //Create string variable str
        std::string str = "Hello, World!";
        std::cout << "String length: " << str.length() << std:: endl;

        //Add characters to end of the string
        str.append(" How are you?");
        std::cout << str << std::endl;

        //Remove character from string str
        str.erase(5, 7);
        std::cout << str << std::endl;

        return 0;
};
```

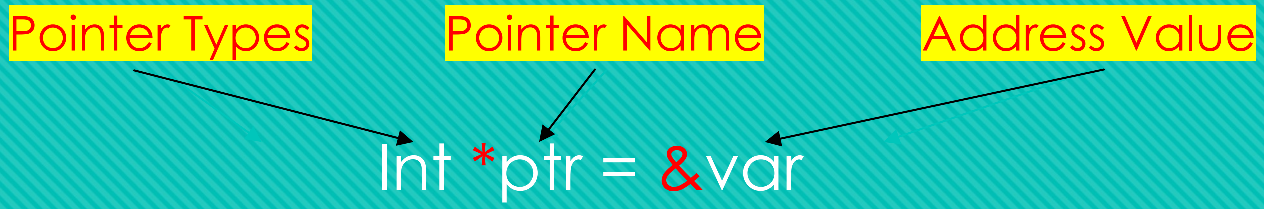


```
ubuntu20@ubuntu:~$ g++ string.cpp -o string
ubuntu20@ubuntu:~$ ls
array                     Desktop          hello.cpp      string
array.cpp                 Documents        ifelse         string.cpp
datainputoutput           dowhileloop      ifelse.cpp     switchcase
datainputoutput.cpp       dowhileloop.cpp  Music          switchcase.cpp
datatype                  Downloads        operators      Templates
datatype.cpp              forloop          operators.cpp  Videos
datatypemodifiers         forloop.cpp      Pictures       whileloop
datatypemodifiers.cpp     hello            Public         whileloop.cpp
ubuntu20@ubuntu:~$ ./string
String length: 13
Hello, World! How are you?
Hello! How are you?
ubuntu20@ubuntu:~$
```

Ref. GeeksforGeeks

3. Pointers

- In C++, a pointer is a variable that stores the memory address of another variable. The operator '&' is used to get the memory address of a variable and the operator '*' is used to access the value stored at the address stored in a pointer. For example:

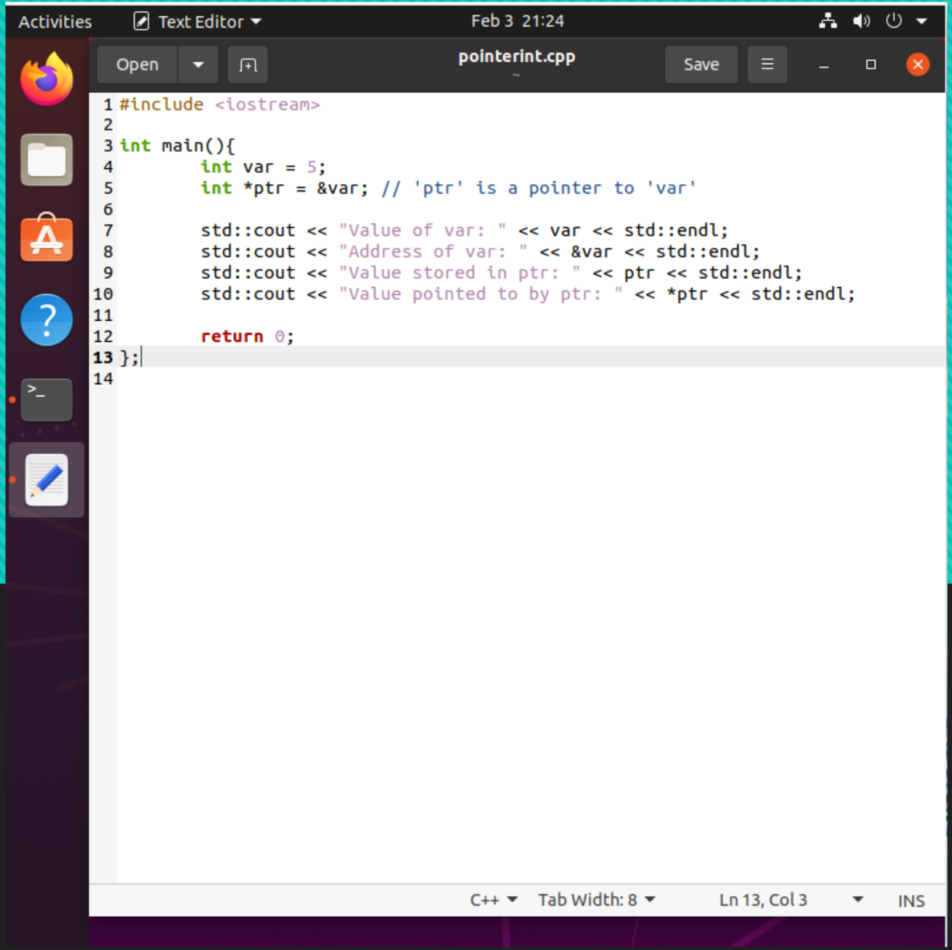


```
Int *ptr = &var
```

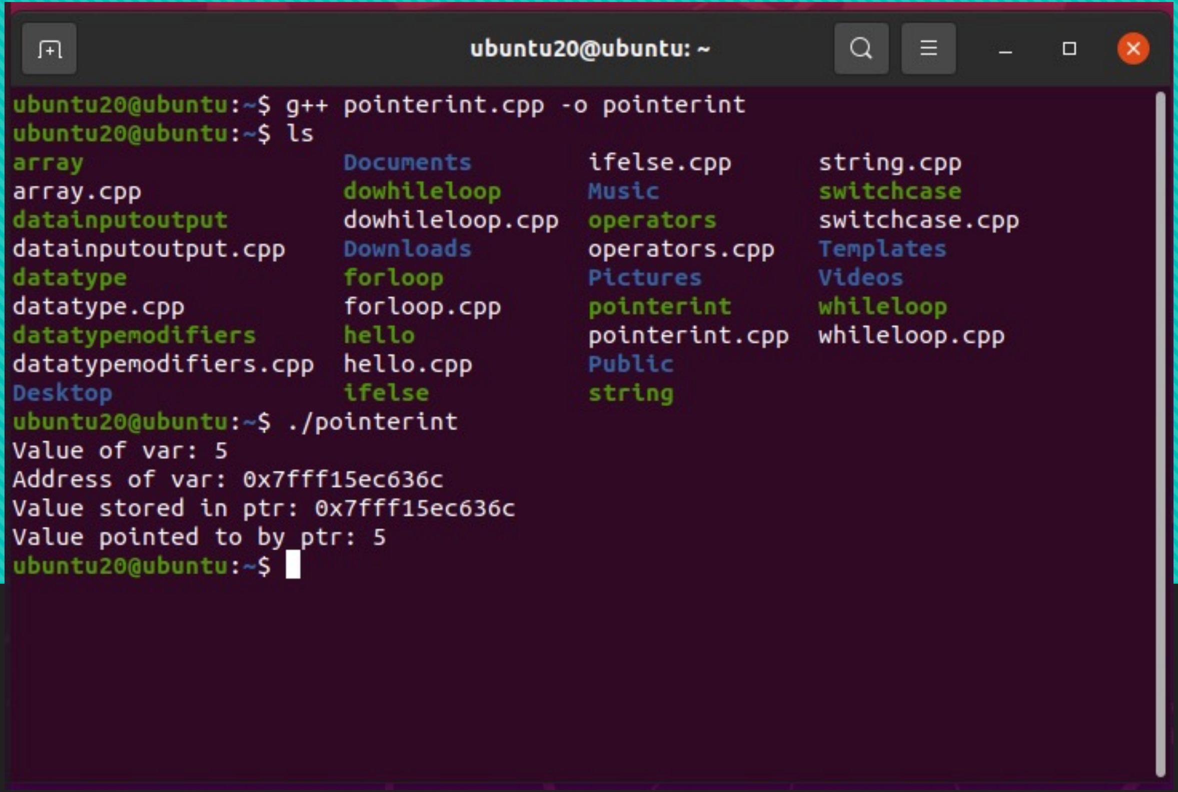
- Pointers are often used to dynamically allocate memory, pass arguments to functions by reference, and for pointer arithmetic. It's important to note that improperly using pointers can lead to undefined behavior and memory leaks, so it's crucial to understand and use pointers correctly in C++.

DASL-100.2
C++ Programming and Linux

3. Pointers



```cpp
#include <iostream>

int main(){
	int var = 5;
	int *ptr = &var; // 'ptr' is a pointer to 'var'

	std::cout << "Value of var: " << var << std::endl;
	std::cout << "Address of var: " << &var << std::endl;
	std::cout << "Value stored in ptr: " << ptr << std::endl;
	std::cout << "Value pointed to by ptr: " << *ptr << std::endl;

	return 0;
};

```

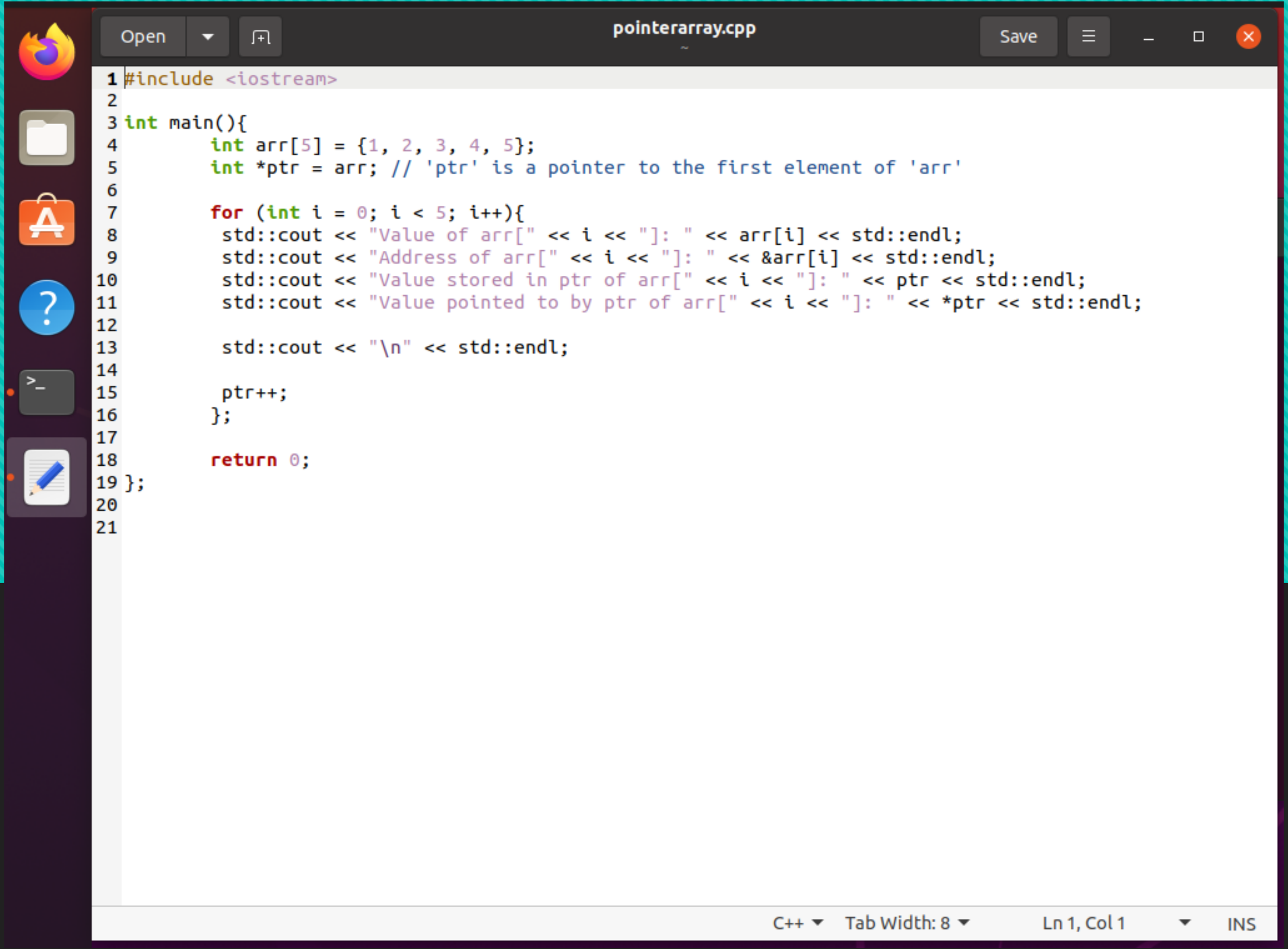


```
ubuntu20@ubuntu:~$ g++ pointerint.cpp -o pointerint
ubuntu20@ubuntu:~$ ls
array                     Documents          ifelse.cpp      string.cpp
array.cpp                 dowhileloop        Music           switchcase
datainputoutput           dowhileloop.cpp    operators       switchcase.cpp
datainputoutput.cpp       Downloads          operators.cpp   Templates
datatype                  forloop            Pictures        Videos
datatype.cpp              forloop.cpp        pointerint      whileloop
datatypemodifiers         hello              pointerint.cpp  whileloop.cpp
datatypemodifiers.cpp     hello.cpp          Public
Desktop                   ifelse             string
ubuntu20@ubuntu:~$ ./pointerint
Value of var: 5
Address of var: 0x7fff15ec636c
Value stored in ptr: 0x7fff15ec636c
Value pointed to by ptr: 5
ubuntu20@ubuntu:~$
```

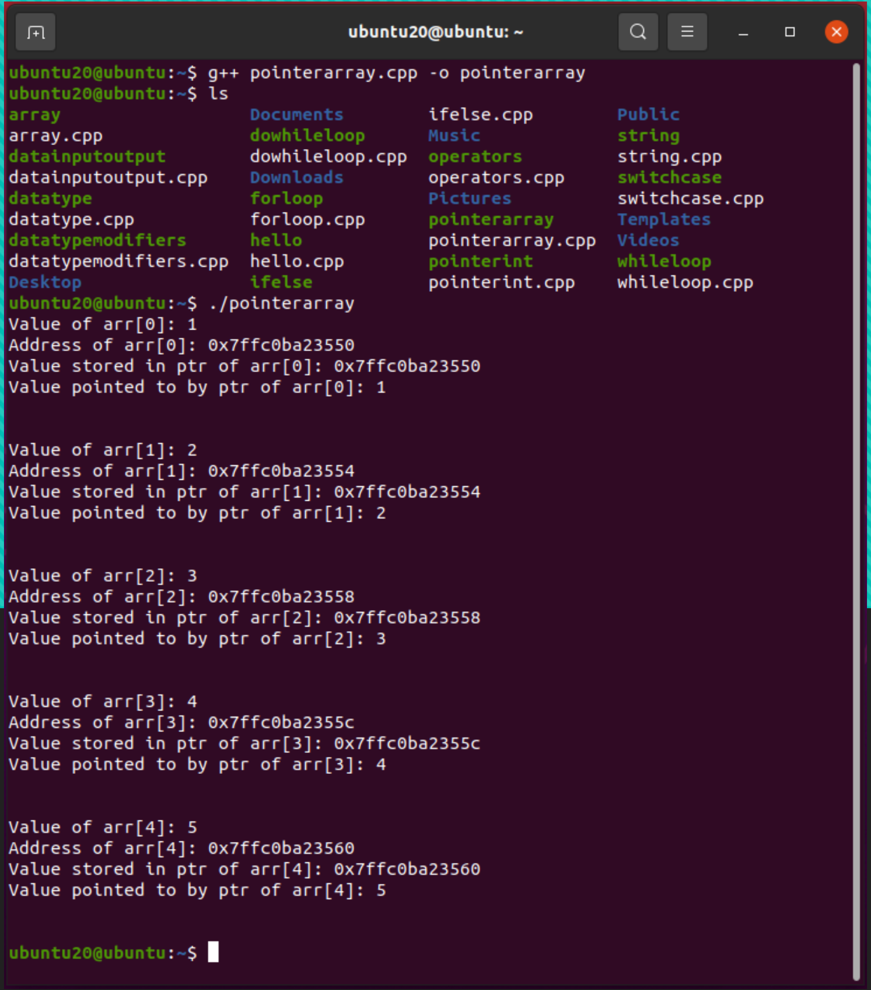
DASL-100.2 C++ Programming and Linux

Drones and Autonomous Systems Lab @ UNLV

3. Pointers



```cpp
#include <iostream>

int main(){
        int arr[5] = {1, 2, 3, 4, 5};
        int *ptr = arr; // 'ptr' is a pointer to the first element of 'arr'

        for (int i = 0; i < 5; i++){
         std::cout << "Value of arr[" << i << "]: " << arr[i] << std::endl;
         std::cout << "Address of arr[" << i << "]: " << &arr[i] << std::endl;
         std::cout << "Value stored in ptr of arr[" << i << "]: " << ptr << std::endl;
         std::cout << "Value pointed to by ptr of arr[" << i << "]: " << *ptr << std::endl;

         std::cout << "\n" << std::endl;

         ptr++;
        };

        return 0;
};


```



```
ubuntu20@ubuntu:~$ g++ pointerarray.cpp -o pointerarray
ubuntu20@ubuntu:~$ ls
array                     Documents        ifelse.cpp        Public
array.cpp                 dowhileloop      Music             string
datainputoutput           dowhileloop.cpp  operators         string.cpp
datainputoutput.cpp       Downloads        operators.cpp     switchcase
datatype                  forloop          Pictures          switchcase.cpp
datatype.cpp              forloop.cpp      pointerarray      Templates
datatypemodifiers         hello            pointerarray.cpp  Videos
datatypemodifiers.cpp     hello.cpp        pointerint        whileloop
Desktop                   ifelse           pointerint.cpp    whileloop.cpp
ubuntu20@ubuntu:~$ ./pointerarray
Value of arr[0]: 1
Address of arr[0]: 0x7ffc0ba23550
Value stored in ptr of arr[0]: 0x7ffc0ba23550
Value pointed to by ptr of arr[0]: 1


Value of arr[1]: 2
Address of arr[1]: 0x7ffc0ba23554
Value stored in ptr of arr[1]: 0x7ffc0ba23554
Value pointed to by ptr of arr[1]: 2


Value of arr[2]: 3
Address of arr[2]: 0x7ffc0ba23558
Value stored in ptr of arr[2]: 0x7ffc0ba23558
Value pointed to by ptr of arr[2]: 3


Value of arr[3]: 4
Address of arr[3]: 0x7ffc0ba2355c
Value stored in ptr of arr[3]: 0x7ffc0ba2355c
Value pointed to by ptr of arr[3]: 4


Value of arr[4]: 5
Address of arr[4]: 0x7ffc0ba23560
Value stored in ptr of arr[4]: 0x7ffc0ba23560
Value pointed to by ptr of arr[4]: 5


ubuntu20@ubuntu:~$
```

3. Pointers

- Pros
  - Pointers provide direct access to memory.
  - Reduces the storage space and complexity of the program.
  - Reduces the execution time of the program.
  - Provides an alternate way to access array elements.
  - Pointers helps us to build complex data structures like linked list, stack, queues, trees, graphs etc.
- Cons
  - Uninitialized pointers might cause segmentation fault.
  - If pointers are updated with incorrect values, it might lead to memory corruption.